VIFOP User Guide: an example of set up and utilization

Gilles Molinié
Francis Auclair
Pôle d'Océanogaphie Côtière de Toulouse,
Laboratoire d'Aérologie, 31400 TOULOUSE
francis.auclair@aero.obs-mip.fr
gilles.molinie@aero.obs-mip.fr

15th December 2003

Contents

1 Introduction

This document intends to show a practical installation, compilation and run of ***VIFOP*** (Variational Initialization and Forcing Platform) . The case study is implemented on a small domain in order to be able to easily play around in tuning the ***VIFOP*** control parameters. The first part of this document looks like a user guide for the installation, compilation and run of ***VIFOP***, the second part describes some optimized field sensitivities to control parameters. ***VIFOP*** is a research oriented toolbox. Thus for users to add developments, some sections consist of technical descriptions of the vector and matrix management in ***VIFOP***, they are labelled "Level 2 Users". A runtime check of ***VIFOP*** is proposed in section **??**.

2 Install the toolbox

2.1 To get the toolbox source files

The source files along the various releases of ***VIFOP*** are managed by ***CVS*** (Concurrent Versions System $http://www.cvshome.org/$), a free source manager. The web browser interface associated to ***CVS*** allows to navigate into the sources and to retrieve any release of any file of the project. To download all the files related to a ***VIFOP*** given release, connect the Noveltis web site: $http://www.noveltis.fr/mfstep-wp7/$, then navigate through out ***WP7-products/How to obtain***. After filling out your login name and password follow the two steps illustrated in figure 1 to select a ***VIFOP*** release and to download it either as a tarball.

2.2 What is included

Untar the downloaded file to create a ***vifop*** directory that can be renamed. The material held in the ***vifop*** directory may not be modified by regular users. The ***VIFOP*** subdirectories include the ***FORTRAN77*** source files (***SOURCES***), the common declarative files (***COMMONS***), the namelists to set ***VIFOP*** input arguments (***NAMELIST***), the files defining the low and high resolution (LR and HR) domains, the LR input fields,... (***LR*** and ***HR***), the mathematical libraries (***LIBRARY***). Several scripts and makefiles are also included in ***vifop*** to install, compile and run ***VIFOP***.

3 Running *VIFOP*

The ***install_vifop*** script creates a ***LOCAL*** directory at the same level as the installed ***VIFOP*** version (currently the ***vifop*** directory). In that ***LOCAL*** directory whose architecture is a copy of ***vifop*** plus a working and a compilation and run diagnostic directories (***WORKDIR*** and ***OUTPUT***), the user can handle the files, for instance modifying or adding source files in the ***SOURCES*** directory, compiling libraries

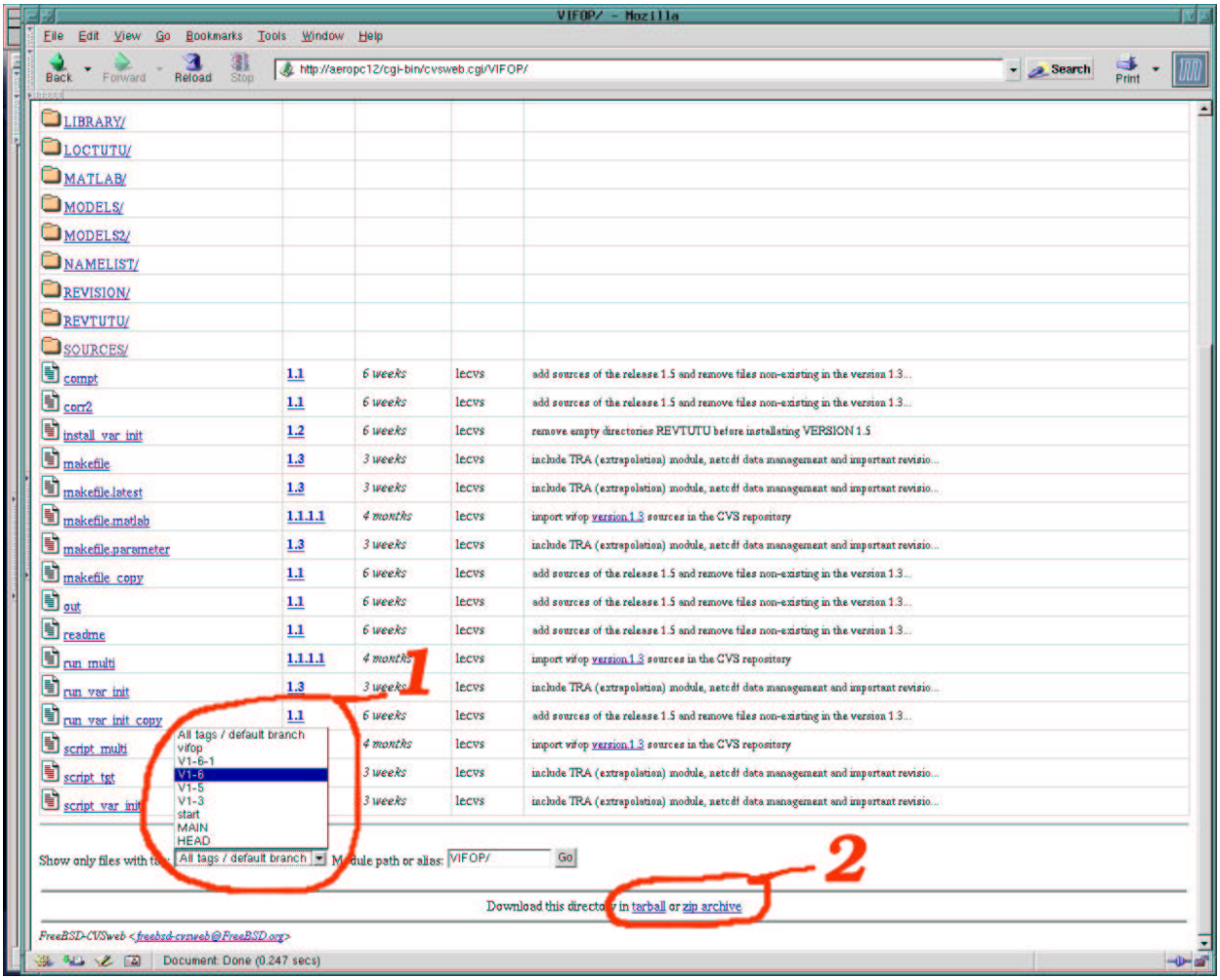

Figure 1: ***VIFOP CVS*** management frontpage. 1: select a ***VIFOP*** release. 2: download it

with different compilers or options,... The ***LOCAL*** directory can be renamed to suit users.

3.1 Install *VIFOP*

The installation procedure is described in the ***INSTALL.TXT*** file of the ***VIFOP*** distribution. The first step is to initialize the shell environment variable ***MAKE*** to the path of your ***gmake*** command. The second step in installing ***VIFOP*** is to set the compiler name and options by editing the ***vifop/script_compil*** shell script (***script_compil*** is the only file required to be edited in ***vifop***). Then, run the ***install_vifop*** script as a shell command. It will be asked where to installed ***VIFOP***, it is assumed in the following that the installation is done in the ***LOCAL*** directory (but no matter wherever it is actually done). It compiles the mathematical libraries considering the chosen compiler name and options and will install the ***LOCAL*** directory at the same level as ***vifop***.

3.2 Compiling and running *VIFOP*

Change directory to ***LOCAL*** and check variables initialized in ***run_var_init***. Insure the ***csckit*** mathematical library is selected (***export LIB_USED='csckit'*** must be set in ***run_var_init***: this is the default). Note that using ***Matlab*** required ***Matlab*** to be installed on your computer.

Then, you are ready to launch the compilation and execution of ***VIFOP*** sources files. A ***PREP_PARAMETER*** step will first be performed in order to define the LR (Low Resolution) and HR (High Resolution) domain parameters, the LR and HR variable lists and the specific parameters to the model configuration for which LR data are optimized. Thus, this first step needs to be done only if you run ***VIFOP*** for the first time or if any of the LR or HR domain has been changed or if the model used is different from this of the previous ***VIFOP*** run. To go through the ***PREP_PARAMETER*** step, launch as a shell command:
$>$ ***run_var_init prep_parameter***
Following the ***PREP_PARAMETER*** step, such a command line compiles and runs the optimization software ***variat_init***.

Compiling and running only ***variat_init*** (in the same domain, and with the same libraries as previously) corresponds to the command line:
run_var_init noprep_parameter.

3.3 Possible compilation or run–time issues

- The objects files may not be linked. Possibly, the mathematical libraries, prep_parameter and variat_init are not compiled with the same compiler or options. To fix this issue, reinstall without removing the ***LOCAL*** directory and recompile the code.

- The optimization can not be proceed. Verify that the correct mathematical libraries are set in the ***run_var_init*** script.
- ...

4 Example case: a walk across *plots.out*

The example case study proposed in the ***VIFOP*** package is called "seamount". The LR and HR bathymetries and the LR fields are included in the ***vifop/LR*** and ***vifop/HR*** directories with the "seamount" suffixes. It represents a seamount at the center of a basin. The data (LR fields) are coarsely discretized ($10 \times 11 \times 7$ points) and must be analyzed on a tighter HR grid ($15 \times 21 \times 11$ points: see fig. 2). The initial fields are not physically consistent.

During the ***VIFOP*** runs, several diagnostic files are written in the ***OUTPUT*** directory. Among them, ***plots.out*** gives an ASCII representation of the ***VIFOP*** arrays, allowing to follow step-by-step the interpolation and optimization process. In the remaining, we scroll through ***plots.out*** as processing the example case. Along the ***VIFOP*** run, several routines write some fields in ***plots.out*** gathered together in blocks[1]. The block header gives the block generic title, the writing routine and the type of the printed fields:

```
****************************************************************
HR grid characteristics                 of type = 1 .
ROUTINE = INTER_INIT_HRGRID.F
****************************************************************
```

The field type indicated here corresponds to one of the four C-grid points[2]. When the displayed field is a vector or a matrix, the type indicates an equation or variable index (this is detailed in the following). The block index is given by the ***OUTPUT*** variable and the field name is indicated one line above. Each value of the 2D arrays are printed out in a table like structure. The *i* index (first index of the 2D arrays) is indicated on the first line and evolves from column-to-column (1...15); the *j* index (second index of the 2D arrays) is printed on the first column and varies from line-to-line (1...21). The same way, vertical and horizontal sections of the 3D variables are plotted. The vertical section indexes can be selected by the ***ISEC_HR_OUT, JSEC_HR_OUT, ISEC_LR_OUT, JSEC_LR_OUT*** variables in ***namelist.output***.

The process to check the installation success is indicated in section 4.3.9.

4.1 Grid descriptions

VIFOP deals with two grids, the ***LR*** (Low Resolution) and the ***HR*** (High Resolution) grid.

[1] Printing a block is activated in ***namelist.output*** through the ***IFL_OUPUT(*)*** flags

[2] Type 1 points are density points, type 2 are $\overline{u}$ points, type 3 are $\overline{v}$ and type 4, ξ vorticity points.

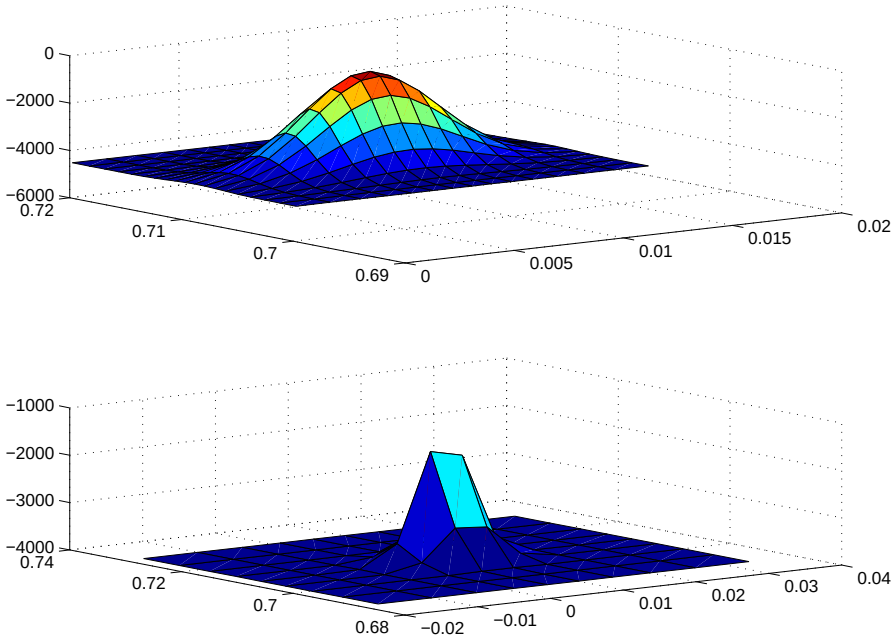

Figure 2: Upper graph: HR (upper graph) and LR (lower graph) discretization of the seamount bathymetry.

4.1.1 The “HR grid characteristics” block

- ***MSK_HR***: Mask of the HR domain
- ***IJ2L_HR***: 1D index for the 2D compressed-vectors stacked representation (see ***VIFOP*** reference manual, section: ”Vector representation”),
- ***H_HR***: Bathymetry,
- ***Lat_HR***: Latitude of grid points,
- ***Lon_HR***: Longitude of grid points,
- ***XM_HR***: West-East coordinate in the Mercator projection,
- ***YM_HR***: South-North coordinate in the Mercator projection,
- ***Z_HR(Surface)***: Height of the sigma-coordinate level near the surface,
- ***Z_HR(Bottom)***: Height of the sigma-coordinate level near the bottom,
- ***Z_HR***: Height of the sigma-coordinate level, ***i*** and ***j*** constant sections,
- ***BEG_HR***: Index for the representation of 3D compressed vectors stacked in 1D vectors (see ***VIFOP*** reference manual, section: ”Vector representation”),

- ***ROT_HR***: grid rotation.

This block is printed out for the 4 point types of the HR C-grids. Sometimes, stars are printed in place of the field value. This is because the conversion of field values (after scaling) into four digit integers, runs over a wide range.

4.1.2 The "LR grid characteristics" block

The printed arrays are similar to those of the HR grid characteristic block.

4.1.3 The "2D and 3D LR-variables" blocks

The 2D and 3D LR-variables are concatenated in the generic arrays called ***V2D_LR*** and ***V3D_LR***. The variable indexes in those generic arrays are set in the ***namelist.lr*** files. These two arrays are printed in the current block. Two horizontal and vertical sections of the 3D variables are printed (the sections are selected via ***namelist.output*** as mentioned above).

4.2 Interpolation

The flag ***IFLINTE*** in ***namelist.flags*** allows selecting the interpolation scheme. ***IFLINTE*** can be set to:

- 1: The gaussian interpolation scheme is processed,
- 2: No interpolation is processed, the all-ready interpolated HR fields are read in the ***$OUT*** [3] ***/restart0.out*** file,
- 3: Copy LR fields in HR variables for model-tangent testing purpose.

The gaussian interpolation of LR data on the HR grid is driven by two parameters (***LBUB_LR*** and ***R_GAUSS***) tunable in the namelist: ***namelist.int***. The flag ***LBUB_LR*** is the radius around HR points defining area in which LR points are selected to be taken into account in the interpolation. The influence of ***LBUB_LR*** on the number of considered LR points is illustrated in fig. 3. Higher **LBUB_LR** values, slower is the interpolation calculation.

Once selected, the data at LR grid-points (Y_{LR}) are interpolated into the HR variable X_{HR} following:

$$X_{HR}(l) = \frac{\sum_k Y_{LR}(k)\ exp(\frac{-D^2(k,l)}{R\_GAUSS^2})}{\sum_k exp(\frac{-D^2(k,l)}{R\_GAUSS^2})} \quad (1)$$

where D(k,l) is the distance between the HR point l and the LR point k (k, l are indexes in the 1D vector representation). The ***R_GAUSS*** parameter allows to adjust the LR-point weights as illustrated in figure 4.

[3] ***$OUT*** is currently the ***OUTPUT*** directory

(a) (b)

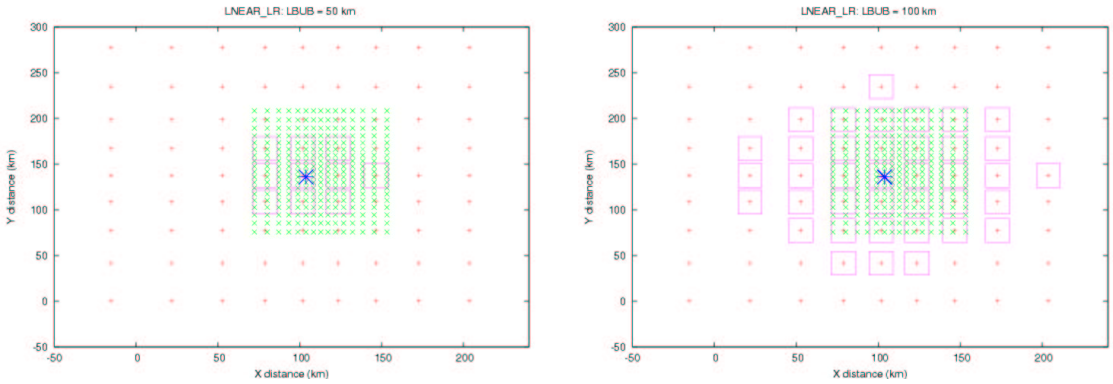

Figure 3: a: the red "+" and the green "x" signs respectively represent the location of the LR grid and HR density-points on a cartesian map. The magenta squares mark the LR selected points to be taken into account in the interpolation of the LR data on the HR density-point highlighted by the blue asterisc in the case of $\boldsymbol{LBUB\_LR} = 50km$; b: same but for $\boldsymbol{LBUB\_LR} = 100km$.

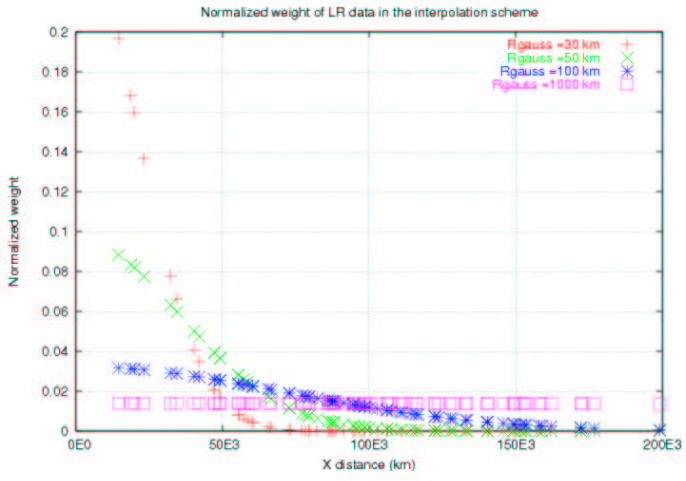

Figure 4: Normalized weight of LR variables in fonction of the distance. Each set of point correspond to a different *R_GAUSS* value.

From an academic sea surface elevation field on the LR grid, fig. 5 shows how can the interpolated fields differ following the ***R_GAUSS*** and ***LBUB_LR*** parameter tuning.

4.2.1 The "Nearest LR stations around LR stations" block (Level 2 Users)

In order to speed up the interpolation scheme, some useful stuffs are processed like the indexes of LR points next to each LR point. These indexes are concatenated in the 1D array ***LNEAR_LR***. The index list beginning for a given LR point L is provided by ***LNEARBEG_LR***(L). Currently, ***LNEARBEG_LR***(L) is displayed in ***plots.out*** but everybody can add the variable he would like to be printed by programming in the file ***output_67.F*** in the section corresponding to ***ICHOICE_P = 45***. For example, ***LNEARBEG_LR LNEAR_LR*** values, currently printed, can be used to verify which points are included in a circle of radius ***LBUB_LR*** around a specific LR point. For instance, for i=5 and j=7, ***LNEARBEG_LR*** indicates 2787 and at i=5 and j=8 ***LNEARBEG_LR***=2876. Then, all the index given by ***LNEAR_LR(2787)*** to ***LNEAR_LR(2876-1)*** indicate the closest LR points to the LR point i=5 and j=7. Their ***i*** and ***j*** indexes could be retrieved from the ***L2I_LR*** and ***L2J_LR*** or ***IJ2L_LR*** arrays, this last one being printed near the ***plots.out*** begin.

"Distance to all LR stations" is also printed in ***plots.out***. This distance is the key parameter to select points that fill out ***LNEAR_LR***. It is printed for a particular point selected in ***namelist.output*** by the ***ISEC_LR_OUT*** and ***JSEC_LR_OUT*** parameters.

This block is repeated for the 4 grid point types.

4.2.2 The "Nearest LR stations around HR grid points" block

Indexes of "Nearest LR stations around HR grid points" are dumped in ***LNEAR_HR***. The way to retrieve LR point indexes is the same as described in the previous section but using ***LNEARBEG_HR*** in place of ***LNEARBEG_LR***. In addition, it is printed the "Distance to all LR stations" from a HR point selected by ***ISEC_HR_OUT*** and ***JSEC_HR_OUT*** in ***namelist.output***.

This block is repeated for the 4 grid point types.

4.2.3 The "Interpolated 2D HR variables" block

V2D and V3D HR-variables resulting of the V2D and V3D LR variable interpolations are displayed in this block.

4.3 Optimization

The optimization leads the model control variables to a compromise between their values involved by the input data and the physical constrains they must verify. The physical constrains are gathered in modulus, the applied modulus can be selected in

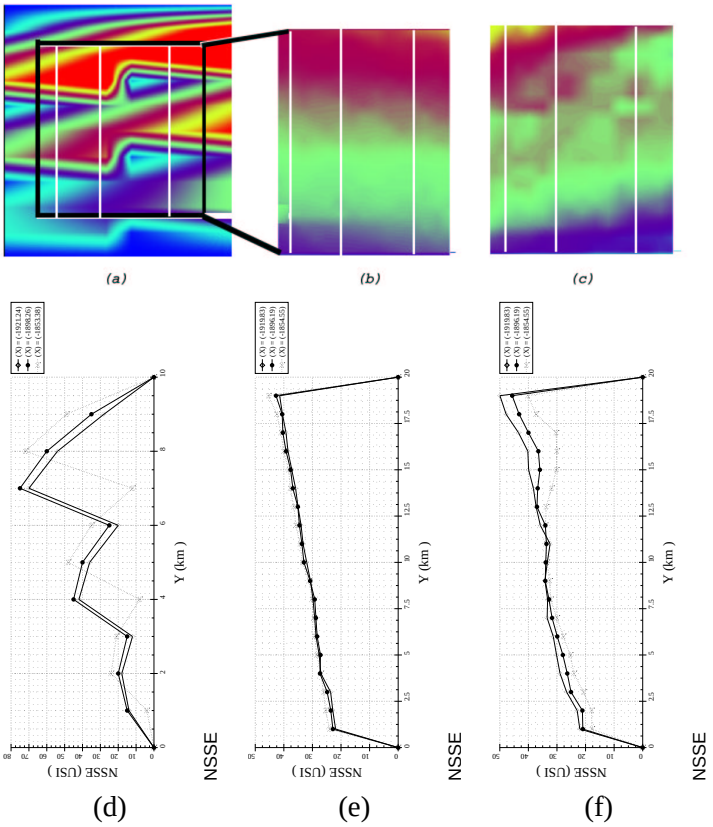

Figure 5: a: LR sea surface elevation anomaly field (NSSE). NSSE ranges from 2 to about 75 cm. The domain size is 300x300 km^2 wide. The black rectangle indicates the HR subdomain ; b: NSSE interpolated on the HR grid with ***LBUB_LR***=50 km and ***R_GAUSS***=50 km; c: same as b but for ***LBUB_LR***=100 km and ***R_GAUSS***=100 km; d: 3 North-South sections of graph (a) are symbolized by thin white lines; e: same as d but sections are from graph (b); f: same as d but section are from graph (c).

namelist.flags. The optimization is proceeded sequentially, that is an optimization modulus computes a set of analyzed variables reintroduced as the first guess in the next modulus. These modulus are :

- ***SSE***: The Sea Surface Elevation modulus reconstructs the sea surface elevation variable,
- ***HPG***: The Horizontal Pressure Gradient modulus constrains the flow to verify the geostrophic balance on Z levels,
- ***MPV***: The Potential Vorticity Modulus constrains the vorticity verifying the quasi-geostrophic balance,
- ***EXT***: This External mode includes constrains on the barotropic mode ($\frac{\partial \eta}{\partial t} = 0$, $\frac{\partial \overline{u}}{\partial t} = 0$, $\frac{\partial \overline{v}}{\partial t} = 0$),

with η the sea surface elevation, $\overline{u}$ and $\overline{v}$ the horizontal west-east and north-south transports. To illustrate the optimization modulus, we will focus on the external mode optimization modulus: the ***EXT*** modulus.

The optimization modulus purpose is to compute the X^a variable such it will be consistent with the model physics to a degree fixed by the constrain covariance errors as shows the cost function equation 2.17 of Auclair *et al.* (2000), R being the constrain covariance matrix and $d = y_0^0 - H_0 X^a$. Each modulus (termed XXXX) computes first y_0^0 and H_0 from a set of equations (constrains) for instance chosen in ***namelist.ext*** for the ***EXT*** modulus by flags ***IFLTC***_XXXX_***EXT***. If both ***IFLTC_DSSEDT_EXT***, ***IFLTC_DUBDT_EXT*** and ***IFLTC_DVBDT_EXT*** are set to 1, the H_0 operator consists on the linear terms of equations 45, 46 and 47 of Blumberg et Mellor (1987) (corresponding respectively to $\frac{\partial \eta(i,j)}{\partial t} = 0$, $\frac{\partial \overline{u}(i,j)}{\partial t} = 0$, $\frac{\partial \overline{v}(i,j)}{\partial t} = 0$) and y_0^0 to the true tendencies $\frac{\partial \eta(i,j)}{\partial t}$, $\frac{\partial \overline{u}(i,j)}{\partial t}$, $\frac{\partial \overline{v}(i,j)}{\partial t}$ obtained after one time step model run (the comprehensive selected constrains, linear and non-linear terms, are coded in ***VIFOP***; $\eta(i,j)$, $\overline{u}(i,j)$ and $\overline{v}(i,j)$ are gathered in the state vector ***X***).

4.3.1 The "External mode variables" block

In order to fill in the state vector ***X***, the external mode variables are retrieved from the V2D and V3D HR-fields. Thus, printed in ***plots.out*** are:

- ***EL_Z***: Sea surface elevation computed at η points of the staggered C grid (as defined in Blumberg et Mellor (1987)),
- ***VBAR_X***: $\overline{u} \times D$ at $\overline{u}$ points ($D = H + \eta$ is the total depth),
- ***VBAR_Y***: $\overline{v} \times D$ at $\overline{v}$ points,
- ***UA_POM***: $\overline{u}$ at $\overline{u}$ points,
- ***VA_POM***: $\overline{v}$ at $\overline{v}$ points,
- ***RHO***: depth averaged density at η points.

4.3.2 The " External mode Smagorinsky dif. coef." block

The block consists of the Smagorinsky diffusion coeficients.

4.3.3 The " Numbering" block (Level 2 Users)

All the model variables (control variables) are dumped into the same 1D state vector (***X***) for each HR-grid points (see the ***VIFOP*** reference manual, section ***X- and Y- vector representation***). At HR point (1,1), type 1, 2, 3, etc non-zero variables are sequentially stored into ***X***. The process to fill in ***X*** continues at (1,2). The tables ***IJ2LX_COM***, ***L2IX_COM*** and ***L2JX_COM*** allows to go back and forth from the 2D to the 1D representation. The ***IJ2LX_COM*** table is shown for the state vector variables in ***plots.out***. ***NUM_ZTA*** corresponds to the numbering of η, ***RHBAR*** to $\overline{\rho}$, ***VBARX*** to $\overline{u}$ and ***VBARY*** to $\overline{v}$. Thus, one can remark that $\eta(1,2)$ is the first value of ***X***, the second is $\overline{\rho}(1,2)$ and the third $\eta(2,2)$ because ***VBARX*** is not defined in the first column (and first row) of the domain. Similarly, ***VBARY*** is not defined in the first 2 rows and in the first column.

4.3.4 The "State Vector Covariance Matrix" block (Level 2 Users)

This is the ***B*** matrix of equation 2.17 of Auclair *et al.* (2000). The way the matrices are stored in memory is explained in the ***Matrix representation*** section of the ***VIFOP*** reference manual. In this block are printed the variables required to retrieve any value of the state vector covariance matrix:

- ***VCOVX_COM***: values of the state vector covariance matrix,
- ***LINCOVX_COM***: beginning of each matrix line,
- ***JCOVX_COM***: non-zero element index in the state vector.

It is very difficult to print matrices that have dimensions up to 3×3 at each grid point. Consequently, only the matrix elements are printed type-by-type [4] in ***VCOVX_COM*** . ***LINCOVX_COM*** indicates the storage index for a new matrix line in ***VCOVX_COM***. Thus, the first element of ***VCOVX_COM*** corresponds to the point (1,2) where ***LINCOVX_COM*** is 1 and as ***LINCOVX_COM*** is 3 at (2,2) one can deduce there are 2 lines in the matrix at (1,2). Moreover, there are 3 lines at the other points of the first row and 4 at the remaining of the domain except column 1. This is consistent with the zero values for ***VBARX*** and ***VBARY*** at the first column and rows as reported in the previous section. ***JCOVX_COM*** indicates the non-zero element indexes in the state vector. The diagonal of ***JCOVX_COM*** being printed type-by-type, consequently, type 1 of ***JCOVX_COM*** is identical to ***NUM_ZTA***, type 2 to ***RHBAR***, type 3 to ***VBAR_X*** and type 4 to ***VBAR_Y***.

[4] the variable type is its index in the state vector ***X***

4.3.5 The "Constrain Matrix" block (Level 2 Users)

This is the ***H*** matrix of equation 2.17 of Auclair *et al.* (2000).

- ***VCONS_COM***: values of the constrain matrix,
- ***LINCONS_COM***: beginning of each matrix line,
- ***JCONS_COM***: non-zero element index in the state vector.

Imagine, we would like to retrieve the $\frac{\partial \overline{u}(i,j)}{\partial t}$ equation terms at (6,3). You must get the index value say
l= ***IJ2LC_COM(6,3,TC_DUBDT_COM)*** and then
$l1$ =***BEGC_COM(L,TC_DUBDT_COM)***. Thus, we have the linearized tendency

$$\frac{\partial \overline{u}(i,j)}{\partial t} = \sum_{k=LINCONS(L1)}^{k=LINCONS(L1+1)-1} VCONS(k) \times X(JCONS(k)) \; .$$

In ***plots.out***, the terms of each selected equation [5] of the optimization modulus are printed. The corresponding blocks are marked by ***"OUTPUT = 150.0000"***. There is first a bunch of blocks associated to each type 1 equation term. The equation type is marked in the block header:

```
************************************************************
Constrain Matrix,                Type= 1
ROUTINE = MODULE_EXT_CONS.F
************************************************************
```

In each block, ***VCONS_COM*** and ***JCONS_COM*** indicate respectively the value and the corresponding state-vector variable index at the point (***ISEC_HR_OUT***, ***JSEC_HR_OUT***) (***namelist.output***) as indicated by the line:

```
Plot of 92 th LIN (I= 8 ,J= 8 ) for Type= 4
```

Thus, the terms printed in the described block correspond to $\overline{v}$ in the equation $\frac{\partial \eta(i,j)}{\partial t} = 0$, that is $\frac{\partial \overline{v}(8,8)}{\partial y}$ which in the POM finite difference scheme is computed between the (8,8) and (8,9) north-south flux points of the C-grid. The indexes reported in ***JCONS_COM*** (be careful to the multiplying factor) are included in ***VBARY*** block 115.

4.3.6 The "External mode variables" block

Here is the beginning of the "red meat". The analysed external-mode variables are printed.

[5] Flags ***IFLTC_DSSEDT_EXT***, ***IFLTC_DSSEDT_ADJ_EXT***, ***IFLTC_DUBDT_EXT*** and ***IFLTC_DVBDT_EXT*** set in ***namelist.ext***, select respectively constrain of type 1 ($\frac{\partial \eta(i,j)}{\partial t} = 0$), type 2 (same as type 1 but applied exactly), type3 ($\frac{\partial \overline{u}(i,j)}{\partial t} = 0$) and type 4 ($\frac{\partial \overline{v}(i,j)}{\partial t} = 0$).

4.3.7 The “After opt.: analyzed 2D or 3D HR variable” block

Display 2D and 3D HR variables recomputed from the analyzed external-mode variables.

4.3.8 The ’Ext. mode analysis’ block

Here begins a bunch of blocks providing analysis of the optimization. The block entitled ***X_ANALYSIS_COM*** display the corrections on the X variables issued from the optimization.

4.3.9 Check *VIFOP* or The “Results of the optimization: Error on constrains” blocks

- ***RHS_COM***: y_0^0 of Auclair *et al.* (2000) equation 2.16,
- ***C_ANALYSIS_COM***: d_0 of Auclair *et al.* (2000) equation 2.16.

If only the ***IFLTC_DSSEDT_ADJ_EXT*** flag has been set to 1 in ***namelist.ext*** (strong constrain, see the end of the current section for a comprehensive description of the incriminated namelists), the optimization result of $\frac{\partial \eta(i,j)}{\partial t} = 0$ must match the ***RHS_COM*** values. Thus,***C_ANALYSIS_COM*** of type 2 must be lower 10^{-4} in the current case using the g77 compiler without specific options). This is a mean to check the routine is well functionning.

namelist.ext must be:

0	IFLTC_DSSEDT_EXT	Optimization of the surface elevation tendency
1	IFLTC_DSSEDT_ADJ_EXT	Optimization of the surface elevation tendency as a strong constrain
0	IFLTC_DUBDT_EXT	Optimization of the depth average x-velocity
0	IFLTC_DVBDT_EXT	Optimization of the depth average y-velocity
0	IFLTC_OBC_VBNORM_EXT	Open boundary conditions for normal depth averaged velocities
0	IFLTC_OBC_VBTGT_EXT	Open boundary conditions for tgt depth
0	IFLTC_SMOOTH_EXT	Optimization of the smoothing constrains

**********External mode constrain covariance**********

1	IFLDIAG_COVC_COM	Flag for a diagonal matrix
0	IFLNLIN_COVC_COM	To compute the matrix from non-linear term estimate
1.e-2	IND_COVC_COM(TC_DSSEDT_EXT)	A-priori error on the d(sse)/dt constrain
2.e-2	IND_COVC_COM(TC_DUBDT_EXT)	A-priori error on the d(ub)/dt constrain
2.e-2	IND_COVC_COM(TC_DVBDT_EXT)	A-priori error on the d(vb)/dt constrain
1.	IND_COVC_COM(TC_SMOOTH_EXT)	A-priori error on the smoothing constrain
0	MATRUNC_COM	Truncation threshold (no threshold = 0.)

**********State vector matrix covariance**********

1	IFLDIAG_COVX_COM	Flag for a diagonal matrix
0	IFLINTER_COVX_COM	To estimate the consequences of the interpolation (1/2/3)
0	IFLBATHY_COVX_COM	To parametrize the difference between the HR and LR bathy
1	IFLHOMO_COVX_COM	To use an homogeneous component

namelist.flags must be:

1	IFLINTE	Flag for the interpolation (1/2/3)
0	IFLINIT_SSE	Flag for the initialization of the SSE
0	IFLINIT_HPG	Flag for the initialization of the truncation error
1	IFLINIT_EXT	Flag for the initialization of the external mode
0	IFLINIT_MPV	Flag for execution of the vorticity modulus
0	IFLINIT_TRA	Flag for execution of the extrapolation modulus

4.3.10 Sensibility tests

Three experiments have been operated. They intend to show the sensibility of the a-priori error on the state vector variables and the constrain sensibilities. Let us first consider only the type 1 constrain ($\frac{\partial \eta(i,j)}{\partial t} = 0$). The ***namelist.ext*** flags are:

1	IFLTC_DSSEDT_EXT	Optimization of the surface elevation tendency
0	IFLTC_DSSEDT_ADJ_EXT	Optimization of the surface elevation tendency as a strong constrain
0	IFLTC_DUBDT_EXT	Optimization of the depth average x-velocity
0	IFLTC_DVBDT_EXT	Optimization of the depth average y-velocity
0	IFLTC_OBC_VBNORM_EXT	Open boundary conditions for normal depth averaged velocities
0	IFLTC_OBC_VBTGT_EXT	Open boundary conditions for tgt depth
0	IFLTC_SMOOTH_EXT	Optimization of the smoothing constrains

Such optimizations will affect $\overline{u}$ and $\overline{v}$. The difference between the first and the second experiment concerns the a-priori error on $\overline{u}$ and $\overline{v}$. In the first experiment, it is set to realistic values:

20.e-2	IND_COVX_COM(TX_SSE_EXT)	A-priori error on the surface elevation
1.5e1	IND_COVX_COM(TX_RHBAR_EXT)	A-priori error on the depth average density (RHb)
20.	IND_COVX_COM(TX_VBARX_EXT)	A-priori error on the x-transport (Ub)
20.	IND_COVX_COM(TX_VBARY_EXT)	A-priori error on the y-transport (Vb)

In the second experiment, the a-priori error on the x- and y-transports are multiplied by 10. The optimized variables after the first and the second experiments are compared to the interpolated fields in fig. 6. We verify that this kind of optimization doesn't affect η. Hence, in the block 165 of ***plots.out***, no modification of the ***X_ANALYSIS_COM*** type 1 variable is reported. Comparisons between the first and the second experiment raise up the role of the a-priori error on the x- and y-transports. This role can also be observed on the ***X_ANALYSIS_COM*** type 3 and 4 variable modification (block 165 of ***plost.out***), that is one order of magnitude larger than in the second experiment. The ***C_ANALYSIS_COM*** values (block 170) reveal that the larger corrections occur around the seamount as ***C_ANALYSIS_COM*** is one order of magnitude lower in the second than in the first experiment around the seamount.

The third experiment considers both the type 3 and 4 constrains ($\frac{\partial \overline{u}}{\partial t} = 0$ and $\frac{\partial \overline{v}}{\partial t} = 0$). The second sensibility test consists of comparing results of the first and the third experiments. The a-priori error are let to realistic values. This comparison is illustrated in fig. 7. In that case, we can see that both ***X*** state vector variables are optimized.

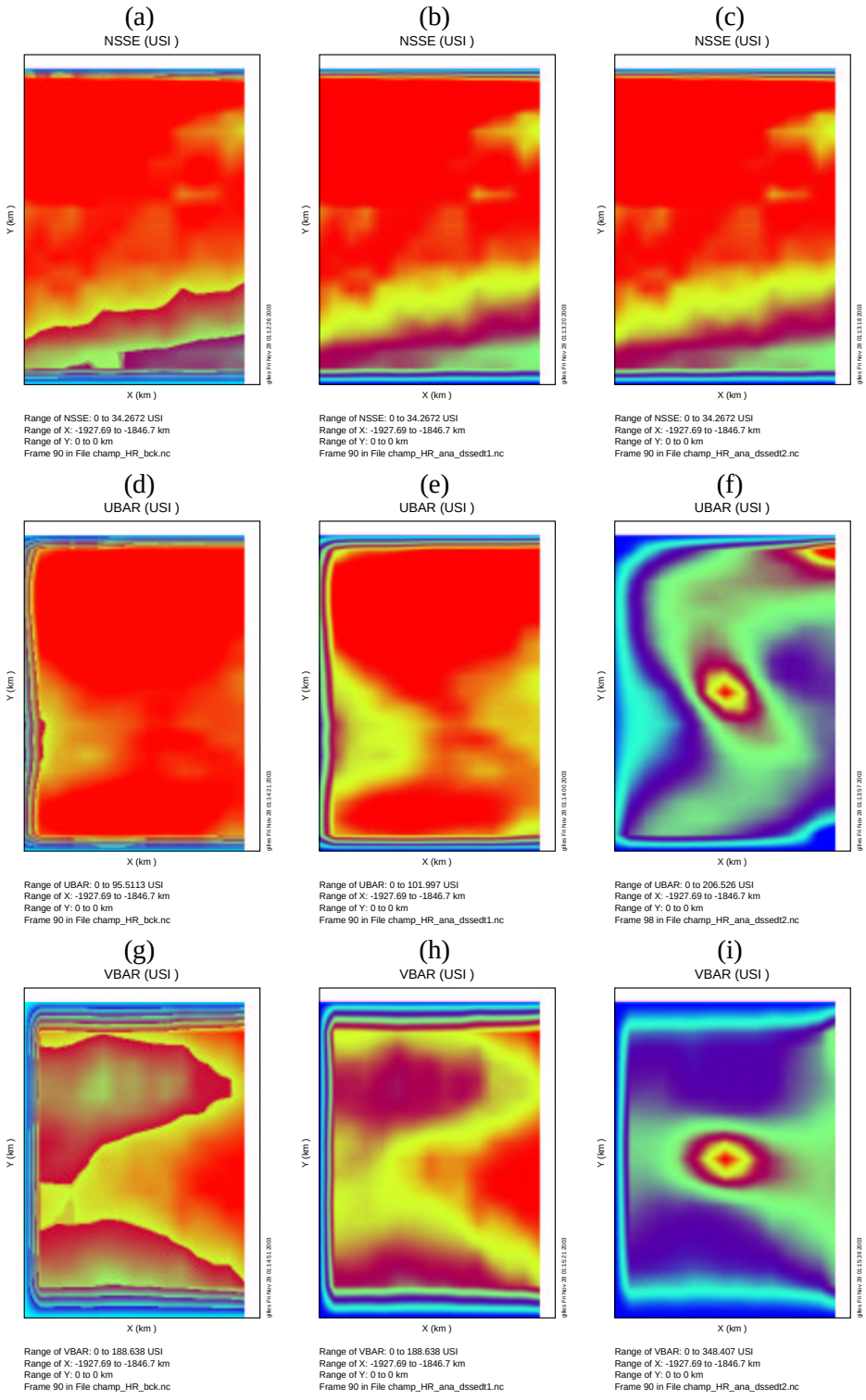

Figure 6: Optimization using the type 1 constrain: $\frac{\partial \eta(i,j)}{\partial t} = 0$. a: Interpolated sea surface elevation; b: same as (a) but optimized; c: same as (b) but large a-priori error on $\overline{u}$ and $\overline{v}$ (that is 200. USI); d: same as (a) but for $\overline{u}$; e: same as (b) but for $\overline{u}$; f: same as (c) but for $\overline{u}$; g: same as (a) but for $\overline{v}$; h: same as (b) but for $\overline{v}$; i: same as (c) but for $\overline{v}$

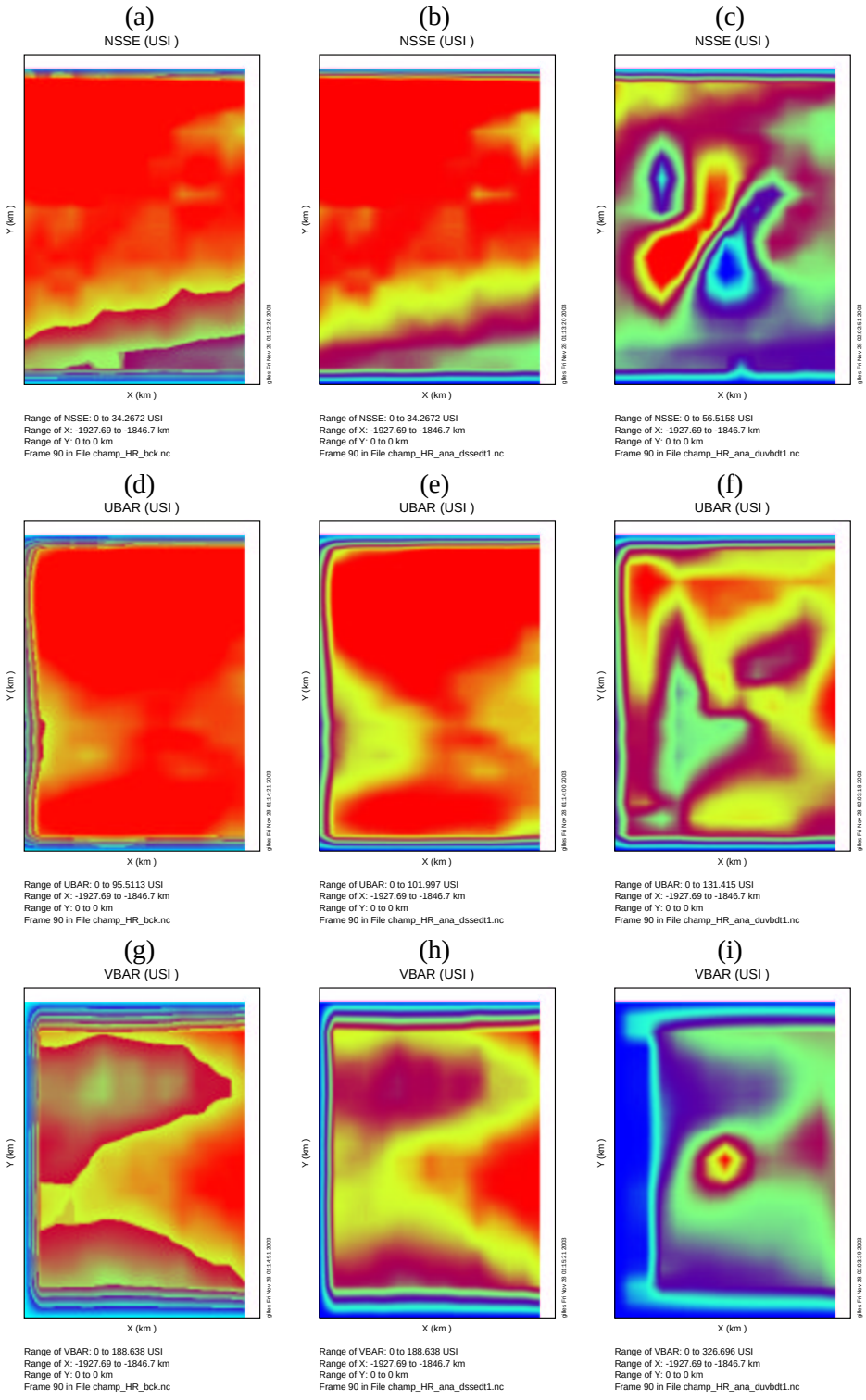

Figure 7: Comparison of the optimization using the type 1 and 2 and 3 constrains. a: Interpolated sea surface elevation; b: same as (a) but for type 1 contrain optimization; c: same as (b) but for type 3 and 4 contrain optimizations ; d: same as (a) but for $\overline{u}$; e: same as (b) but for $\overline{u}$; f: same as (c) but for $\overline{u}$; g: same as (a) but for $\overline{v}$; h: same as (b) but for $\overline{v}$; i: same as (c) but for $\overline{v}$

References

Auclair, F., S. Casitas, and P. Marsaleix, 2000: Application of an inverse method to coastal modelling. *J. Atmos. Oceanic Technol.*, **17**, 1368–1391.

Blumberg, A. F. and G. L. Mellor, 1987: *A description of a three-dimensional costal ocean circulation model*, volume 4 of *Three-Dimensional Costal Ocean Models*. American Geophysical Union.